개발문서

AR 인공지능 게임 프로그램 개발

머드게임 "Way To Signiel" 제작 문서

한창신

목차

1. 게임소개

1-1. 게임소개

주어진 초기자본 5천만원으로 경마에 베팅하여 목표금액 30억을 달성하면 승리하는 게임이다. 경마는 총 5개로 구성되어 있으며, 플레이어는 원하는 경마에 돈을 베팅한다. 각 마는 정해진 확률로 한번씩 이동하게 되며, 확률에 따라 1칸, 3칸, 혹은 5칸을 이동한다. 각 마의 칸 수 이동 확률은 다르게 설정되어 있으며, 이에 비례하여 배당률도 다르게 설정되어 있다. 마가 이동해야 하는 칸은 총 50칸이며, 한 마가 목표 칸을 달성했을 시 경기는 종료된다. 플레이어가 베팅한 마가 승리하면 해당 마의 배당률에 따라 돈을 받는다.

1-2. 진행 방식

베팅: 플레이어는 보유한 금액 안에서 자유롭게 베팅할 수 있으며, 중복 베팅도 허용한다.

확률 아이템: 플레이어는 보유한 금액 안에서 자유롭게 아이템을 구매할 수 있지만, 중복구매는 허용하지 않는다. 아이템은 1칸 이동 확률을 내리고 3칸이나 5칸 이동 확률을 높이는 방식으로 되어 있다.

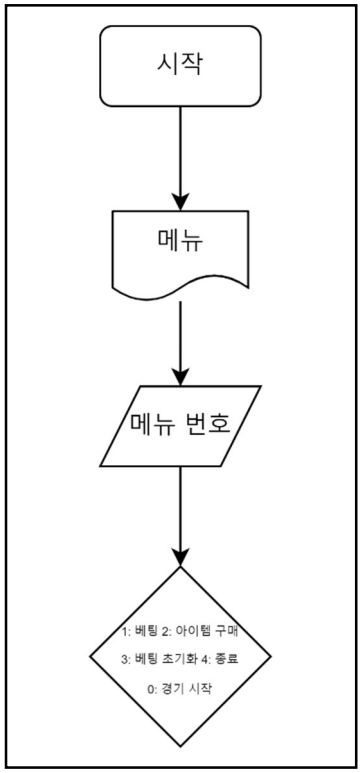
베팅 초기화: 플레이어가 마에 베팅한 모든 금액을 초기화하여 0으로 만든다. 베팅한 금액은 플레이어 자본으로 반환된다.

게임종료: 게임을 종료한다. 내용은 저장되지 않는다.

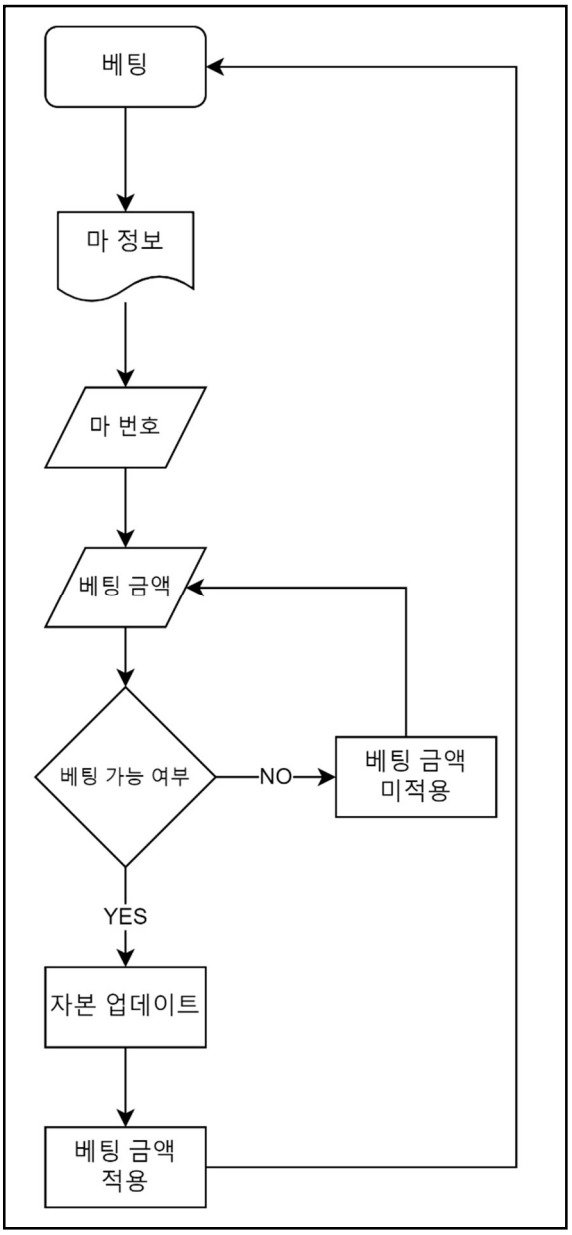
게임시작: 경마가 시작되며, 플레이어는 약 10초간 경마 경기를 시청한다. 경기가 종료되면 플레이어가 승리 마에 베팅 여부에 따라 승/패 내용이 출력되며, 플레이어의 돈이 상승하거나 유지된다..

2. 순서도

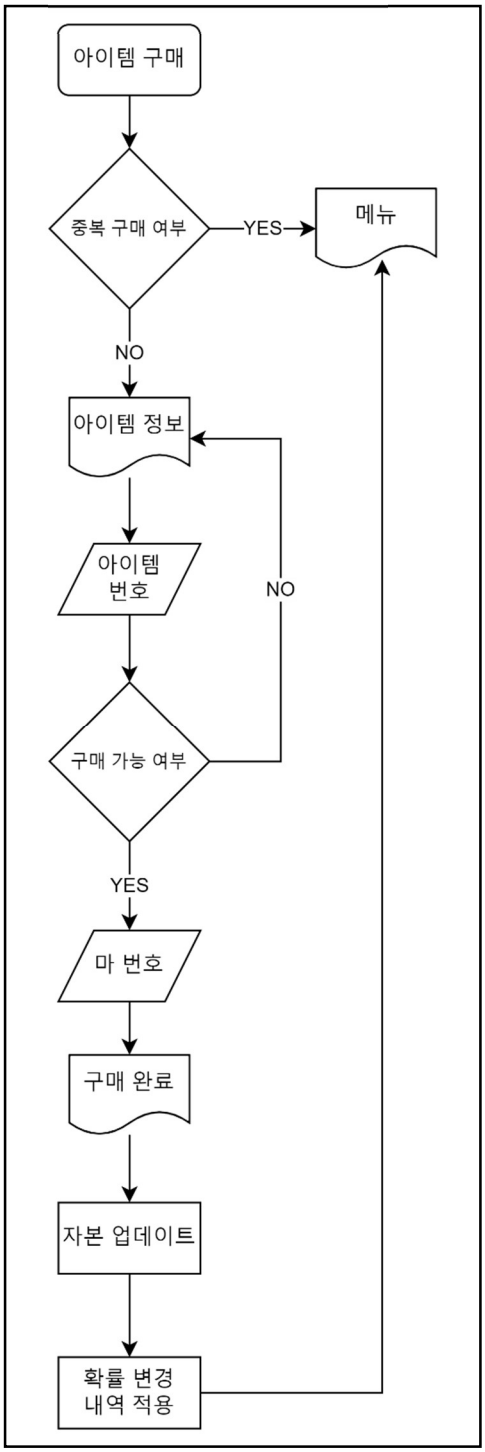
2-1. 시작 순서도



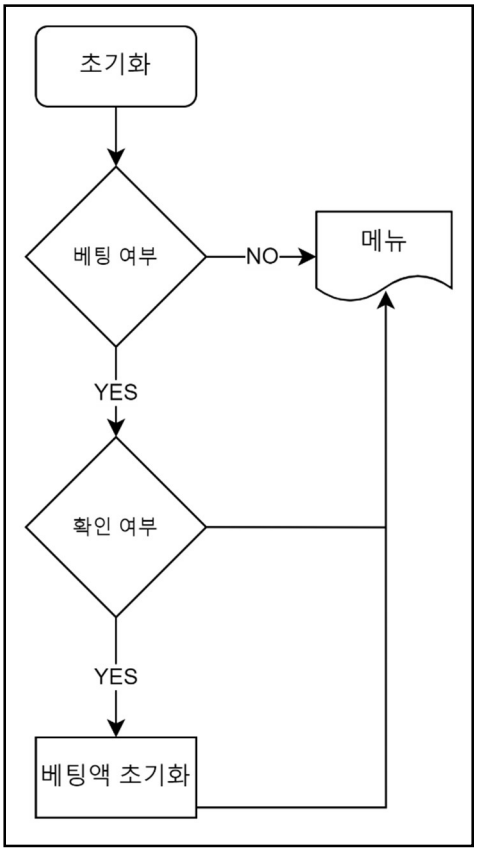
2-2. 베팅 순서도



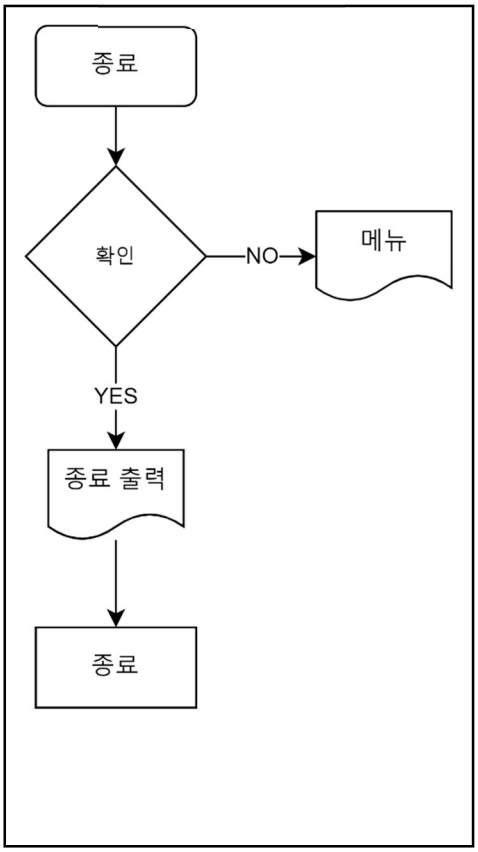
2-3. 아이템 구매 순서도



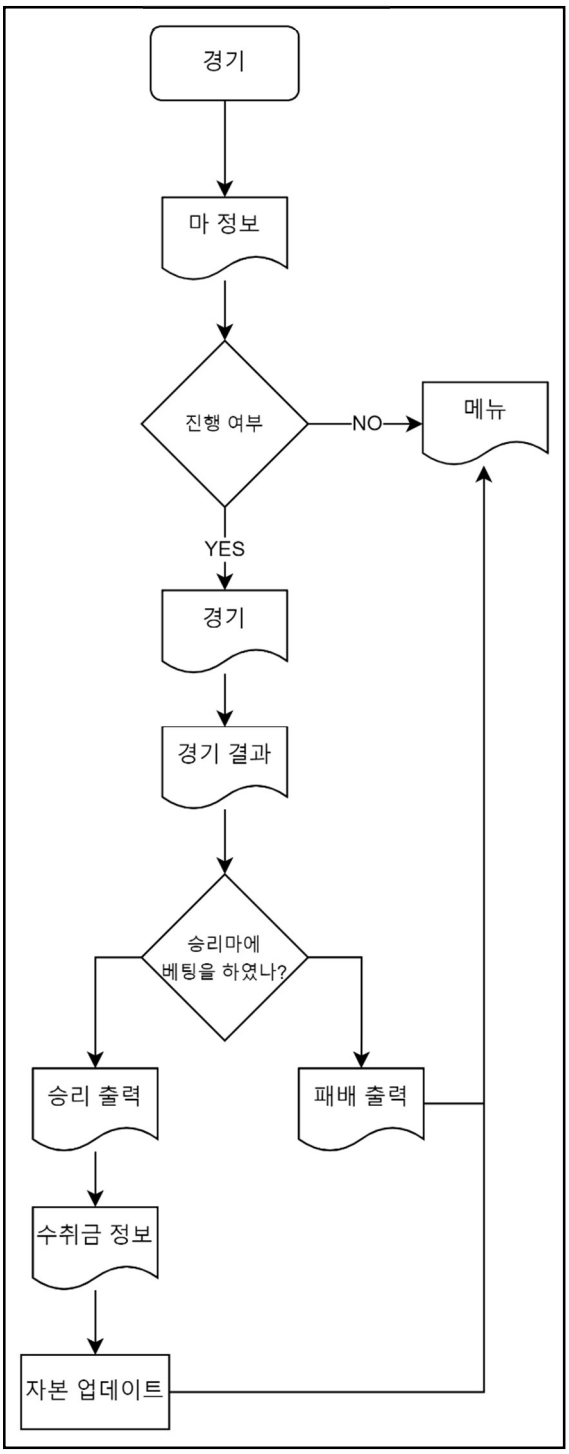
2-4. 초기화 순서도



2-5. 종료 순서도



2-5. 경기 순서도



3. 헤더파일

3-1. define

GM: 원활한 테스트 진행을 위해 건너뛰기, 승률 조작 등 진행
MUSIC_PATH: 음악 파일 폴더 경로
INTRO_SKIP: 게임 오프닝, 처음에만 진행되는 화면들 설정
DIST: 경마 경기 거리, *2로 폭으로도 사용됨 (경마 길이 특수문자로 *2)
SEC_STEP: 경마가 한번 가는데 딜레이 시간
WORD_SPEED: 한 문자 씩 나오는 출력 속도
PATH: 경마 경기 시 마가 지나지 않은 길 아이콘
PASSED: 경마 경기 시 마가 지난 길 아이콘
RESET: 커서 좌측 상단으로 이동 (커서 초기화)
DOWN: 커서 개행
UP: 커서 한 줄 상승
START_MONEY: 시작 자본 설정
TARGET_MONEY: 목표 자본 설정

3-2. 전역변수

int money: 보유 금액 변수
int before_money: 경기 결과 나오기 전 금액 변수
str[300]: 출력 문자열 임시 저장 변수

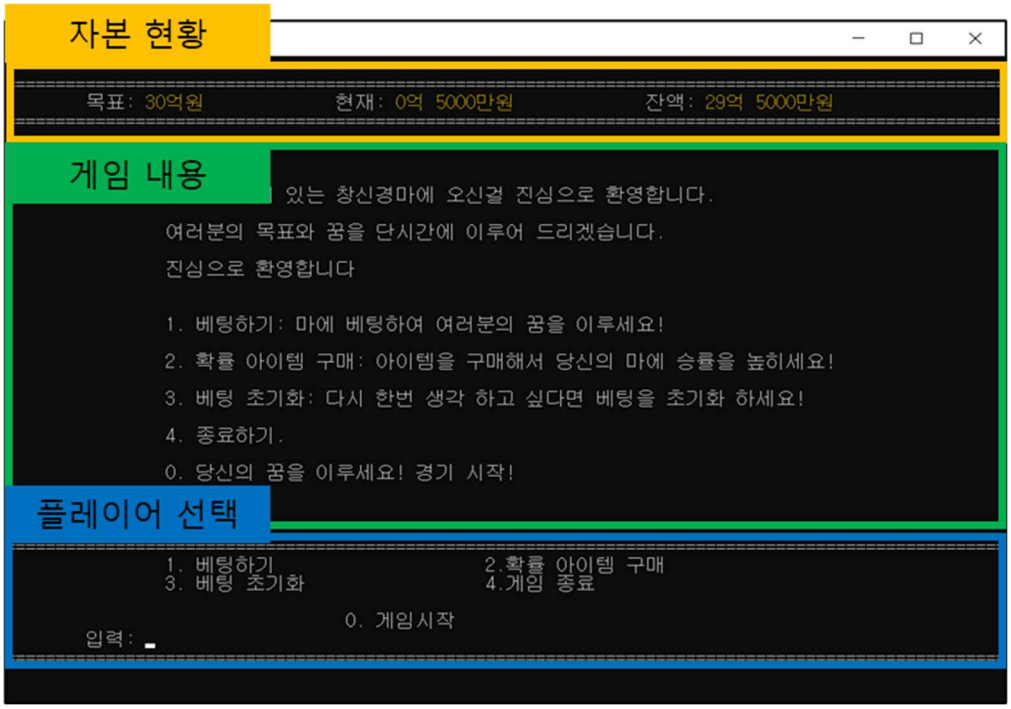
3-3. 구조체

horses: 각 마의 구조체로 마 번호, 이름, 아이콘, 칸 이동 확률, 배당률, 베팅 금액 등 설정

4. 함수

4-0. 구분

- 화면구분



- 함수 소스파일 구분

1. general_functions.c: 특정 상황에서 사용되는 것이 아닌 어떤 상황에서도 사용되는 함수들
2. rae_functions.c: 경마 경기 진행에 필요한 함수들
3. bet_functions.c: 마 정보 및 경마에 베팅하기 위해 필요한 함수들
4. iteam_functions.c: 아이템 구매에 필요한 함수들
5. prints.c: 문자열 출력 집합 함수들로 특정 상황으로 함수들을 묶어놓았다.

4-1. 일반적 함수

- 화면을 구분하는 선 출력

```
void box_line()
{
    printf("₩033[0m₩n");
    for (int i = 0; i < DIST * 2; i++)
    {
        printf("=");
    }
    DOWN;
}
```

- 커서가 게임내용 끝 부분에 있을 시 플레이어 선택 화면 초기화

```
void resetwords_top_bot()
{
    box_line();
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < DIST * 2; j++)
            printf(" ");
        }
        DOWN;
    }
    box_line();

    for (int i = 0; i < 6; i++)
    {
        UP;
    }
}
```

- 커서가 플레이어 선택 화면 아래 있을 때 플레이어 선택 화면 초기화

```
void resetwords_bot_top()
{
    for (int i = 0; i < 8; i++)
    {
        UP;
    }
    box_line();
    for (int i = 0; i < 5; i++)
    {
        for (int j = 0; j < DIST * 2; j++) {
            printf(" ");
        }
        DOWN;
    }
    UP;
    box_line();
    for (int i = 0; i < 6; i++)
    {
        UP;
    }

}
```

- 화면에 한 문자씩 출력되는 함수

```
void printw(char* input)
{
    char str[500];
    strcpy(str, input);

    for (int i = 0; i < strlen(str); i++)
    {
        putchar(str[i]);
        Sleep(WORD_SPEED);
    }
}
```

ㅇ (매개변수) char* input: 문자열을 입력받는다.

ㅇ 설명: 한 문자씩 출력되는 함수를 printf함수와 유사하게 사용하고자 해서 생성한 함수로 printf와 동일하게 매개변수 안에 문자열을 넣으면 된다.

- 플레이어 선택 화면에서 정보를 플레이어가 받은 후 엔터를 눌러 다음으로 넘어가게 하는 함수

```
void pressenter()
{
    char* str = "press enter";

    for (int i = 0; i < DIST * 2 - strlen(str)-10; i++)
    {
        printf(" ");
    }
    printf("%s", str);
    _getch();
}
```

- 자본 현황 출력. (money는 전역 변수로 매개변수로 받지 않고 사용)

```
void printmoney()
{
    box_line();
    printf("Wt목표:W033[33m %d억원WtWt W033[0m현재: W033[33m%d억 %d만원WtWtW033[0m잔액: W033[33m%d억 %d만원W033[0m",W
    get_uk(TARGET_MONEY), get_uk(money), get_man(money), get_uk(TARGET_MONEY - money), get_man(TARGET_MONEY-money));

    box_line();
    DOWN;
}
```

- 어떤 방식으로든 게임이 끝나면 출력되는 마지막 문자열

```
void goodbye()
{
    char* str = "WtWt최고의 주사위 던지기는WnWtWtWt주사위를 통에 그냥 넣어두는 것이다.WnWnWnWtWtWtWt감사합니다. 제작자. 한창신.";
    system("cls");
    for (int i = 0; i < 5; i++) DOWN;
    for (int i = 0; i < strlen(str); i++)
    {
        putchar(str[i]);
        Sleep(70);
    }
    Sleep(3000);
}
```

- 게임 시작시 오프닝 출력 함수

```
void printstart()
{
    FILE* fp;
    char str[8][100] = { 0 };

    fp = fopen("text.txt", "r");
    if (fp == 0)
    {
        printf("text.txt 열기 실패");
        exit(1);
    }

    system("cls");
        for (int i = 0; i < 8; i++)
        {
            fgets(str[i], 100, fp);
        }

        for (int i = 0; i < 100; i++)
        {
            for (int k = 0; k < 8; k++)
            {
                for (int j = 0; j < i; j++)
                {
                    if (k == 3 && str[k][j] == '-') printf("\033[33m%c", str[k][j]);
                    else printf("\033[0m%c", str[k][j]);
                }
                if (i < strlen(str[k])) DOWN;
            }

            for (int j = 0; j < 8; j++)
            {
                UP;
            }

            Sleep(10);
        }

        for (int i = 0; i < 8; i++)
        {
            DOWN;
        }

        for (int i = 0; i < 6; i++)
        {
            fgets(str[i], 100, fp);
        }
        for (int i = 0; i < 100; i++)
        {
            for (int k = 0; k < 6; k++)
            {
                for (int j = 0; j < i; j++)
                {
                    if (k == 3 && str[k][j] == '-') printf("\033[33m%c", str[k][j]);
                    else printf("\033[0m%c", str[k][j]);
                }
                if (i < strlen(str[k])) DOWN;
            }


            for (int j = 0; j < 6; j++)
            {
                UP;
            }

            Sleep(10);
        }

        for (int i = 0; i < 12; i++)
        {
            DOWN;
        }
    printf("\t\t\t\t\t\t\t\t\t\t\t\t\t\t   제작. 한창신");
    Sleep(3000);
    printf("\n\n\n\n");
    {
        char* str = "\033[90m— press enter to start —";

        for (int i = 0; i < DIST+18 - strlen(str); i++)
        {
            printf(" ");
        }
        printf("%s", str);
        _getch();
    }
    fclose(fp);
}
```

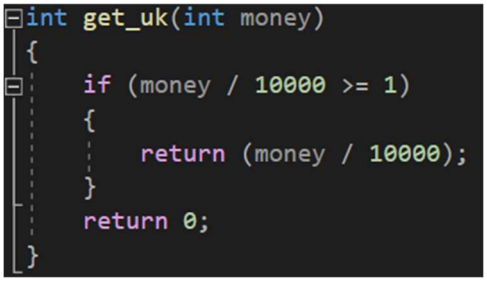
- 정수 2 개를 매개변수로 두어서 최소, 최대값을 정의하고 scanf 로 입력 받는 함수

```
int input(int min, int max)
{
    int operation;

    printf("\n");
    while (1)
    {
        printf("\t입력: ");

        scanf("%d", &operation);

        if (operation >= min && operation <= max)
        {
            DOWN;
            break;
        }
        else
        {
            DOWN;
            printf("                                \r");
        }
    }

    return operation;
}
```

ㅇ매개변수:

int min: 입력 받고자 하는 정수의 최소값 / int max: 입력 받고자 하는 정수의 최대값

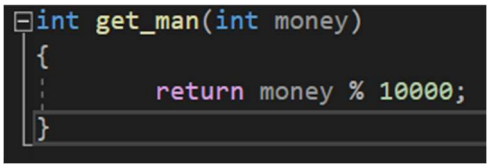
- get_uk(int money): 돈을 매개변수로 받아 억 단위를 반환하는 함수



```
int get_uk(int money)
{
    if (money / 10000 >= 1)
    {
        return (money / 10000);
    }
    return 0;
}
```

ㅇ 매개변수: int money: 억 단위를 반환하고자 하는 돈을 받는다.

- get_man(int money): 돈을 매개변수로 받아 억 단위를 제외한 천 단위를 반환하는 함수



```
int get_man(int money)
{
        return money % 10000;
}
```

ㅇ 매개변수: int money: 억 단위를 제외하고 천 단위를 반환하고자 하는 돈을 받는다.

- 목표 금액을 달성하면 출력되는 함수.

```
void game_clear()
{
    FILE* fp;
    char str[18][100] = { 0 };

    system("cls");

    fp = fopen("building.txt", "r");
    if (fp == 0)
    {
        printf("text.txt 열기 실패");
        exit(1);
    }

    for (int i = 0; i < 18; i++)
    {
        fgets(str[i], 100, fp);
    }

    for(int i=0; i<18;i++)
    {
        printf("%s", str[i]);

        Sleep(250);
    }
    Sleep(1000);
    printf("\n\n");


}
```

- 경기가 끝난 후 소유 자본이 0 이면 실행되는 게임 오버 및 다시 시작 확인 함수

```
int game_over()
{
    int operation;
    system("cls");
    Sleep(1000);
    PlaySound(TEXT("D:\\HCS\\2023\\first_mudgame\\first_mudgame\\music\\gameover"), NULL, SND_FILENAME | SND_ASYNC);
    printw("\n\n\n\n\t.... 당신은 모든 돈을 잃으셔서 창신경마장에서 쫓겨나셨습니다.\n\t\t\t\t\t\t플레이해주셔서 감사합니다.\n\n\n");
    Sleep(3000);
    printw("\t\t다시 플레이 하시겠습니까?\n\n\t1. 네\n\t2.아니요\n");

    operation = input(1, 2);
    Sleep(2000);

    return operation;
}
```

반환값: opreation 을 입력 받아 반환 후 다시 시작여부 결정한다.

- 음악 재생 파일 및 반복 재생 설정 함수

```
void play_music(const char* file_name, int b_loop)
{
    char path[100] = { 0 };
    strcpy(path, MUSIC_PATH);
    strcat(path, file_name);
    strcat(path, ".wav");

    int len = strlen(path) + 1;
    int size = MultiByteToWideChar(CP_UTF8, 0, path, len, NULL, 0);
    wchar_t* w_path = (wchar_t*)malloc(size * sizeof(wchar_t));
    MultiByteToWideChar(CP_UTF8, 0, path, len, w_path, size);

    if(b_loop==0)
    PlaySound(w_path, NULL, SND_FILENAME | SND_ASYNC);
    else if(b_loop==1)
    PlaySound(w_path, NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
}
```

ㅇ 매개변수:

const char* file_name: 파일 이름 문자열 입력 / b_loop: 반복 재생 여부 입력(1:반복, 0: 미반복)

4-2. 경기 진행 함수

- 마 정보 입력 받는 함수

```
void set_horses(struct horses* horse, int num, const char* name, const char* imo, int per1, int per3, int per5, int percent)
{
  //확률 합이 100이 아닐 경우 -> exit
  if (per1+per3+per5!=100)
  {
    printf("%s의 확률합이 100이 아닙니다.", name);
    exit(1);
  }

  horse->num = num;
  horse->name = name;
  strcpy(horse->imo, imo);
  horse->per_1 = per1;
  horse->per_3 = per3;
  horse->per_5 = per5;
  horse->percent = percent;

}
```

ㅇ 매개변수:

struct horses* horse: 정보를 입력하고자 하는 마 입력

int num, const char* name, const char* imo: 해당 마의 번호, 이름, 아이콘 입력

int per1, per3, per5: 1 칸, 3 칸, 5 칸 갈 확률 입력

- 마 정보 시작 설정으로 초기화 하는 함수

```
void reset_horses(struct horses* horse)
{
    set_horses(&horse[0], 1, "이상해씨", "☎", 50, 25, 25, 2);
    set_horses(&horse[1], 2, "키타산블랙", "※", 55, 20, 25, 4);
    set_horses(&horse[2], 3, "까미 냥이", "◎", 60, 20, 20, 6);
    set_horses(&horse[3], 4, "CY.NIRO ", "◆", 65, 15, 20, 8);
    set_horses(&horse[4], 5, "우리 우디", "◐", 70, 15, 15, 10);
}
```

ㅇ 매개변수:

struct horses* horse: 구조체 배열의 주소를 입력 받아 선언된 구조체의 배열 전체 설정

ㅇ설명: 게임시작시 마 정보 선언 및 아이템 사용 후 칸 이동 확률 초기화를 위해 사용

- 경기 실행시 각 마의 이동 칸수 확률 무작위 진행

```c
int rand_step(struct horses horse)
{
    int dice;

    dice = rand() % 100;
    horse.per_1 -= 1; //0부터이기 위해
    if (dice <= horse.per_1) return 1;
    else if (dice <= horse.per_1 + horse.per_3) return 3;
    else if (dice <= horse.per_1 + horse.per_3 + horse.per_5) return 5;
}
```

ㅇ 매개변수: struct horses horse: 마의 각 칸 이동 확률을 불러오기 위해 구조체 입력

ㅇ 반환값: 무작위 뽑기 진행 후 해당 마가 1, 3, 5 칸 중 이동 할 수 반환

- 마 경기 진행 함수

```c
void go_horse(struct horses* horse, int step, int start, int* finish, int* winner)
{
    if (start == 1) step = 1; //시작시 스타트 라인에 한줄로 서있기 위해
    if (*finish == 1) step = 0; //경기 완료 시 아래 함수들이 실행 안되기 위해

    int go = 0;
    printf("\033[0m\n  %d번 %s    베팅금: %d\n", horse->num, horse->name, horse->tot_bet);

    for (int k = 0; k < step; k++) //step 만큼 반복: 삭제->출력
    {
        if (k != 0)
        {
            //한줄 삭제하기 (특수문자는 칸을 2칸 차지하나봄 ㅎ)
            for (int i = 0; i < DIST; i++) printf("\b\b");
        }

        for (int i = 0; i < DIST; i++) //한줄 만들기
        {
            if (horse->tot_steps + go == i) printf("\033[32m%s", horse->imo);
            else if (i < horse->tot_steps + go) PASSED;
            else PATH;

            if (horse->tot_steps + go >= DIST - 1)
            {
                *finish = 1;
                *winner = horse->num-1;
            }
        }
        go++;
        if (*winner != 100) break;

        Sleep(SEC_STEP / step); //움직이는 속도 조정 (몇 칸을 가든 소모되는 시간은 동일함)
    }
    printf("\n\n");
    horse->tot_steps += step;
}
```

ㅇ 매개변수:

struct horses* horses: 이동 차례가 된 마 입력

int step: 해당 마가 rand_step 에서 반환받은 칸 이동 수

int start: 시작시 스타트 라인에 한줄로 서있기 위해 시작 첫 함수 실행 여부 확인

int* finish: 목적지 도착 완료 및 다른 마가 먼저 도착 시 이동하지 않기 위한 변수

int* winner: 목적지에 먼저 도착한 승리 마 번호 받기 위한 변수

ㅇ 설명:

경기 시작 후 해당 함수가 첫 실행이 아니고, 아직 목적지에 도착한 마가 없을 때 해당 마가 반환 받은 이동 칸 개수 만큼 앞으로 이동할 수 있는 함수. 마가 3 칸이나 5 칸 이동 시 한번에 이동하는 것이 아닌 1 칸과 같은 시간 내에 3 칸, 5 칸을 이동하게 한다. 어떤 마가 목적지에 도착 시 마의 번호를 winner 변수 포인터로 받고, 다른 마들은 더 경기를 진행하지 않기 위해 함수를 바로 탈출 할 수 있도록 finish 변수가 있다.

- 경기 시작시 해설자 내용 및 카운트 다운 출력 함수

```
void start_race()
{
    resetwords_top_bot();
    char str[300] = "안녕하십니까 여러분, 창신 경마에 참여해주셔서 감사합니다.\
\n이제 곧 경기가 시작 됩니다.\
\n행운을 빕니다.\n";
    for (int i = 0; i < strlen(str); i++)
    {
        putchar(str[i]);
        Sleep(WORD_SPEED);
    }
    for (int i = 3; 0 < i; i--)
    {
        Sleep(300);
        printf("%d....", i);
        Sleep(300);
    }
}
```

- 경기 종료 후 해설자 내용 및 승리마 서식지정자로 출력

```
void finish_race(struct horses horse)
{
    resetwords_top_bot();

    //40번째: 38(번호), 45번째: m (이름)
    char str[300] = "경기가 끝났습니다!! 승자는 바로바로.... 번마 입니다!!\
\n맞추신 분에게는 축하를, 아쉬운 분들은 다음에 꼭 맞추시길 기원하며,\
\n다음 경기 때 뵙겠습니다!\
\n감사합니다 :)\n";
    for (int i = 0; i < strlen(str); i++)
    {
        if (i == 38) str[i] = horse.num + '0';
        if (i == 45) printf("\033[33m<%s %s %s>\033[0m", horse.imo, horse.name, horse.imo);
        putchar(str[i]);
        Sleep(WORD_SPEED);
    }
}
```

ㅇ 매개변수: 승리한 마 매개변수 입력하여 승리마 서식지정자로 번호 및 입력 출력

- 경기 종료 후 승리 마에 베팅한 돈이 0 원 이상일 시(=수취금이 있을 시) 출력되는 내용

```
void if_wins(struct horses horse)
{

    //32: 마 번호, 38: 마 이름, 42: 베팅액 73: 배당률 98: 수령금
    char str[300] = "진심으로 축하드립니다.!! \n\
당신은  번마 에 억 만원\033[0m을 베팅 하셨고, 배당률은  배 입니다.\n\
총 수령 금액은 억  만원\033[0m 입니다. 다시 한번 진심으로 축하드립니다!!!\n\n";

    for (int i = 0; i < strlen(str); i++)
    {
        if (i == 33) str[i] = horse.num + '0';
        if (i == 39) printf("\033[33m%s%s%s\033[0m", horse.imo, horse.name, horse.imo);
        if (i == 42) printf("\033[33m%d", get_uk(horse.tot_bet));
        if (i == 45) printf("%d", get_man(horse.tot_bet));
        if (i == 78) printf("\033[33m%d\033[0m", horse.percent);
        if (i == 105) printf("\033[33m%d", get_uk(horse.tot_bet * horse.percent));
        if (i == 108) printf("%d", get_man(horse.tot_bet * horse.percent));
        putchar(str[i]);
        Sleep(WORD_SPEED);
    }
}
```

o 매개변수: 승리 마를 매개변수로 입력받아 승리 마에 베팅한 금액, 배당률, 수취금 등 내용 출력

- 수취금이 있을 시 현재 돈에서 수취금을 더한 돈까지 자본현황에서 올라가는 함수

```
void printmoneyrace()
{
    int speed = (money - before_money) / 101;
    box_line();
    box_line();
    printf("\033[F");
    printf("\033[F");
    play_music("cash1", 0);
    while (speed > 1)
    {

        printf("\t목표:\033[33m %d억원\t\t \033[0m현재: \033[33m%d억 %04d만원\t\t\033[0m잔액: \033[33m%d억 %d만원\033[0m", \
            get_uk(TARGET_MONEY), get_uk(before_money), get_man(before_money), get_uk(TARGET_MONEY - before_money), get_man(TARGET_MONEY - before_money));
        Sleep(20);
        printf("\r");
        for (int i = 0; i < DIST * 2; i++)
        {
            printf(" ");
        }
        printf("\r");
        before_money += speed;
        if (before_money >= money - speed) break;
    }
    printf("\t목표:\033[33m %d억원\t\t \033[0m현재: \033[33m%d억 %d만원\t\t\033[0m잔액: \033[33m%d억 %d만원\033[0m", \
        get_uk(TARGET_MONEY), get_uk(money), get_man(money), get_uk(TARGET_MONEY - money), get_man(TARGET_MONEY - money));
    box_line();
    printf("\n");
    Sleep(500);
    play_music("cash2", 0);
    Sleep(2000);
}
```

o 설명: 승리 마에 0 원 이상 베팅 시 베팅금*배팅률이 현재 자본과 더해져서 새로운 자본이 되는데, 새로운 자본까지 올라가는 것을 시각적으로 부각하기 위한 함수. 수취금을 101 로 나눈 후 새로운 자본과 같을 때 까지 101 번 더한다.

- 승리마에 베팅한 돈이 0 원일 때 실행되는 출력 함수

```
void if_lose()
{
    char str[300] = "아쉽지만 승리 마를 맞추지 못하셨습니다.\n\
다음에는 꼭 더 좋은 기회가 있을 것입니다.\n\
너무 실망하지 마시고, 다음에는 꼭 승리하시길 기원하며,\n\
다음에 뵙겠습니다!!\n";
    for (int i = 0; i < strlen(str); i++)
    {
        putchar(str[i]);
        Sleep(WORD_SPEED);
    }
}
```

4-3. 베팅 함수

- 베팅 할 마 선택 시 해당 마의 이름과 내용 출력

```
void ask_bet(struct horses horse)
{
    printf("\t%d번 %s에 얼마를 베팅하시겠습니까?(단위: 만원)\n\n\n", horse.num, horse.name);
}
```

ㅇ 매개변수: 플레이어가 베팅을 하기 위해 선택한 마를 매개변수로 사용

- 베팅 금액 입력 후 베팅 가능 여부 확인하는 함수

```
void bet_input(struct horses* horse)
{

    ask_bet(*horse);
    int bet_total;
    //resetwords_bot_top();

    while (1)
    {
        horse->bet = input(-999999, 999999);

        bet_total = horse->tot_bet + horse->bet;

        if (money - horse->bet >= 0 &&bet_total >= 0 && bet_total <= 999999) break;
        if (bet_total <= 0)
        {
            resetwords_bot_top();
            printf("\t베팅액이 음수가 될 수 없습니다. 다시 입력 바랍니다: \n\n\n");
        }
        if (money - bet_total < 0)
        {
            resetwords_bot_top();
            printf("\t빚을 져서 베팅을 할 수 없습니다. 도박중독 상담 : 1336번\n\n\n");
        }
    }
}
```

ㅇ 매개변수: 플레이어가 베팅하고자 하는 마를 매개변수로 받는다.

ㅇ 설명: 베팅금을 입력하기 위해서는 아래와 같은 내용을 확인 후 조건이 충족한다면 베팅을 진행한다.

1. 베팅 최소, 최대 금액 범위 안에 있는가, 2. 베팅 후 플레이어의 자본이 0 과 같거나 이상인가, 3. 해당 마의 베팅 금액이 0 과 같거나 이상인가.

- 마 정보를 표로 출력 할 때 행 구분 줄 출력

```
void inform_line()
{
    printf("\t");
    for (int i = 0; i < DIST * 2 - 19; i++)
    {
        printf("-");
    }
    DOWN;
}
```

- 마 정보를 표로 출력 할 때 각 마 정보 출력 함수

```
void horseinform(struct horses horse)
{
    printf("\t|%3d   | %-10s | %5d    | %5d    | %5d    |   x %2d  | %d 만원\t|\n", \
        horse.num, horse.name, horse.per_1, horse.per_3, horse.per_5, horse.percent, horse.tot_bet);
    inform_line();
}
```

ㅇ 매개변수: 출력하고자 하는 마 입력

ㅇ 설명: 마 정보를 horse.num 을 기준으로 1 번부터 5 번까지 출력되는데, 출력되는 마의 정보를 출력하는 함수

4-4. 아이템 구매 함수

- 아이템 구매 후 마에 적용하는 함수

```
void steroid(struct horses* horse, int per_3, int per_5)
{
    horse->per_1 -= per_3 + per_5;
    horse->per_3 += per_3;
    horse->per_5 += per_5;
}
```

ㅇ 매개변수:

struct horses* horse: 플레이어가 아이템을 적용한 마를 입력받는다.

int per_3, int per_5: 플레이어가 구매한 아이템의 3 칸, 혹은 5 칸 상승률을 받아 더해주고, 1 칸 이동 확률에는 해당 수치만큼 빼준다.

4-5. 문자열 함수

- 각종 출력내용 함수

ㅇ 설명: main 소스파일에 출력해야 할 문자열을 최소화 하기 위해 소속이 없는, 베팅시, 경기시, 리셋시, 게임 끝났을 시 등 6 가지 상황으로 구분하여 main 문에서 사용할 때 함수를 호출하는 방식을 채택하였다.

```
char* print_general(int num)
{
    switch (num)
    {
    case 0:
        //메인화면 출력
        strcpy(str, "\n\n\t\t꿈과 희망이 있는 창신경마에 오신걸 진심으로 환영합니다.\n\n\
\t\t여러분의 목표와 꿈을 단시간에 이루어 드리겠습니다.\n\n\
\t\t진심으로 환영합니다\n\n\n\
\t\t1. 베팅하기: 마에 베팅하여 여러분의 꿈을 이루세요!\n\n\
\t\t2. 확률 아이템 구매: 아이템을 구매해서 당신의 마에 승률을 높히세요!\n\n\
\t\t3. 베팅 초기화: 다시 한번 생각 하고 싶다면 베팅을 초기화 하세요!\n\n\
\t\t4. 종료하기.\n\n\
\t\t0. 당신의 꿈을 이루세요! 경기 시작!\n\n\n");
        break;

    case 1:
        //메인화면 선택 출력
        strcpy(str, "\t\t1. 베팅하기\t\t\t2.확률 아이템 구매\n\t\t3. 베팅 초기화\t\t\t4.게임 종료\n\n\t\t\t\t\t  0. 게임시작");
        break;

    case 2:
        //게임 클리어
        strcpy(str, "\n\t\t\t\t\t\t진심으로 축하드립니다\n\n\n\t당신은 5천만원을 30억으로 만드시는데 성공했습니다.\n\t게임에서와 같은\
행운이 항상 당신과 함께하길 기원합니다.\n\t\t\t\t\t플레이 해주셔서 감사합니다.\n\n");
        break;

    case 3:
        //제작자 이름
        strcpy(str, "\n\n\t\t\t\t\t\t\t\t       제작. 한창신");
    }

    return str;
}
```

```
//race
char* print_race(int num)
{
    switch (num)
    {
    case 0:
        //베팅 확인
        strcpy(str, "\n\t아래와 같은 금액을 배팅하셨습니다.\n\n\t1번을 눌러 시작하세요.\n\n\t당신의 승리가 한발짝 앞에 있습니다.\n\n");
        break;

    case 1:
        //경기 진행 컨펌
        strcpy(str, "\t이대로 진행하시겠습니까 ?\n\n\t1. 네, 제 직감은 틀림 없습니다.\n\t2. 아니요, 잠깐만요");
        break;
    }

    return str;
}

//bet
char* print_bet(int num)
{
    switch (num)
    {
    case 0:
        //bet 메인화면
        strcpy(str, "\t창신경마에 오신걸 환영합니다.\n\n\t모든 경마는 1칸/3칸/5칸 중 하나를 아래와 같은 확률로 이동합니다.\n\t경마를 선택 후 배팅액 입력 후 경마를 시작해주세요. \n\n\t승리는 당신의 것입니다.\n\n");
        break;

    case 1:
        //마 정보 구분 행
        strcpy(str, "\t| 번호 |     이름     | 1칸 확률 | 3칸 확률 | 5칸 확률 |  배당율 |    배팅액    |\n");
        break;
    }

    return str;
}
```

o 매개변수:

int num: 해당 문자열의 집합에서 출력하고자 하는 문자열 번호를 입력한다.

5. 메인